Logistic Regression

Introduction

Oh!, a wine factory is going to receive a new pack of different wines and they do not have their type labelled (red or white). Ok, don't worry, we can go through each of the wines, look at it color, and label it. But... we would like to do this process automatically. In the following lines, we will face a classification problem to predict if the wine is red or white, depending on its physicochemical attributes.

A classification problem relates input variables x to the output variable y, but now y can take only discrete values, instead of continuous variables as in regression. When y can only take two discrete, it is called binary classification. We will denote these values as $y \in \{0, 1\}$ in the rest of the report, where $0 \equiv$ white class and $1 \equiv$ red type.

Loading the data

The data science pipeline often* needs to split the original dataset into two smaller pieces: the train and test datasets. If we only evaluate our models in the same dataset, the results will be overestimated (aka overfitting). To provide honest assesments of the performance of the predictive models, we will need to validate the models using a test dataset, a partition that has not been used to build the models in order to avoid bias.

(*) Some statistical learning models are robust enough to do not need this division. They can infer the behaviour of the whole population from a sample if some statistical hypothesis are fulfilled.

```
# Loading the dataset into a dataframe
df <- read_delim("../data/processed/wines.csv", ";", escape_double = FALSE, trim_ws = TRUE)

# Train and test dataset, split 80%.
split = nrow(df)*0.8
train = df[1:split,]
test = df[split:nrow(df),]
```

In this case, the test dataset consists of the 20% of the dataset (1300 observations).

Logistic Regression Model

The equivalent linear regression model in classification is the logistic regression model. This model needs to specify a function such that $p(y = 0|\tilde{\boldsymbol{X}})$ and $p(y = 1|\tilde{\boldsymbol{X}})$ are both greater than 0 and sum 1. The logistic function has such properties, defininig the following model:

$$p(y|\tilde{\boldsymbol{X}}, \boldsymbol{\beta}) = \frac{e^{\boldsymbol{\beta}\tilde{\boldsymbol{X}}}}{1 + e^{\boldsymbol{\beta}\tilde{\boldsymbol{X}}}}$$

If $\beta_i > 0$ then increasing one unit in x_i will increase the probability of a success. If $\beta_i < 0$, then the probabilty of success decrease when increasing x_i. When $\beta_i = 0$, $e^0 = 1$, so the odds do not change with x_i.

Full model

We start by defining a logistic regression model with all the 11 attributes as the predictors. We do not use the `quality`, used in regression, and neither the `type`, used as the target variable:

```r
# Logistic regression model with all the variables.
log.full=glm(type~fixed_acidity+volatile_acidity+citic_acid+residual_sugar+chlorides
             +free_sulfur_dioxide+total_sulfur_dioxide+density
             +pH+sulphates+alcohol, data=train, family=binomial)
summary(log.full)
```

```
##
## Call:
## glm(formula = type ~ fixed_acidity + volatile_acidity + citic_acid +
##     residual_sugar + chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     density + pH + sulphates + alcohol, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.4910  -0.0567  -0.0230  -0.0002   5.5097
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -2.427e+03  2.284e+02 -10.628  < 2e-16 ***
## fixed_acidity        -9.649e-01  2.529e-01  -3.815 0.000136 ***
## volatile_acidity      5.047e+00  1.110e+00   4.545 5.50e-06 ***
## citic_acid           -2.111e+00  1.364e+00  -1.548 0.121606
## residual_sugar       -9.974e-01  1.105e-01  -9.022  < 2e-16 ***
## chlorides             2.035e+01  4.152e+00   4.901 9.55e-07 ***
## free_sulfur_dioxide   7.211e-02  1.396e-02   5.164 2.41e-07 ***
## total_sulfur_dioxide -5.504e-02  5.522e-03  -9.967  < 2e-16 ***
## density               2.436e+03  2.320e+02  10.498  < 2e-16 ***
## pH                   -4.644e+00  1.589e+00  -2.922 0.003475 **
## sulphates             1.592e+00  1.382e+00   1.152 0.249313
## alcohol               2.782e+00  3.569e-01   7.797 6.36e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5843.79  on 5195  degrees of freedom
## Residual deviance:  331.46  on 5184  degrees of freedom
## AIC: 355.46
##
## Number of Fisher Scoring iterations: 9
```

At first sight, each of the coeffiecient has a marginal test which attempts the null hypothesis H_0: $\beta_i = 0$, after adjusting the coefficients within the model. That means, it is checked the net effect of each variable and whether should be in the model or not. All the p-values are small enough to reject H_0 (considering $\alpha = 0.05$) except for `citric_acid` (0.12) and `sulphates` (0.249). Let us discard these two variables in the further analysis:

```r
# Logistic regression model with all the variables.
log.F=glm(type~fixed_acidity+volatile_acidity+residual_sugar+chlorides
             +free_sulfur_dioxide+total_sulfur_dioxide+density
             +pH+alcohol, data=train, family=binomial)
summary(log.F)
```

```
##
## Call:
```

```
## glm(formula = type ~ fixed_acidity + volatile_acidity + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     density + pH + alcohol, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.6691  -0.0555  -0.0226  -0.0004   5.4365
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -2.529e+03  2.094e+02 -12.080  < 2e-16 ***
## fixed_acidity        -1.102e+00  2.389e-01  -4.614 3.95e-06 ***
## volatile_acidity      5.677e+00  9.761e-01   5.816 6.02e-09 ***
## residual_sugar       -1.042e+00  1.003e-01 -10.385  < 2e-16 ***
## chlorides             1.894e+01  4.094e+00   4.627 3.72e-06 ***
## free_sulfur_dioxide   7.492e-02  1.379e-02   5.435 5.49e-08 ***
## total_sulfur_dioxide -5.620e-02  5.409e-03 -10.390  < 2e-16 ***
## density               2.538e+03  2.127e+02  11.930  < 2e-16 ***
## pH                   -4.596e+00  1.524e+00  -3.015  0.00257 **
## alcohol               2.922e+00  3.201e-01   9.129  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5843.79  on 5195  degrees of freedom
## Residual deviance:  335.69  on 5186  degrees of freedom
## AIC: 355.69
##
## Number of Fisher Scoring iterations: 9
```

All the marginal tests are now significantly small and the overall fit of the model is high enough ($p = 1$), so we do not have evidence to reject the model in favour of the simple constant model.

```
pchisq(log.F$deviance, log.F$df.residual, lower=F)
```

```
## [1] 1
```

Validation

We will define the following function to validate the correctness of the model. It basically counts the number of correctly classified observations, and it is divided by the total number of examples.

```
accuracy <- function(model, train, test){
  # Train error
  train_prob=model$fitted
  train_prob=ifelse(train_prob>0.5,1,0)
  d_train = table(train_prob, train$type)

  # Test error
  test_prob = predict(model, newdata = test, type = "response")
  test_prob = ifelse(test_prob>0.5,1,0)
  d_test = table(test_prob, test$type)

  train_accuracy = sum(diag(d_train))/sum(d_train)
```

```
  test_accuracy = sum(diag(d_test))/sum(d_test)

  return(list(train_accuracy, test_accuracy))
}
```

```
accuracy(log.F, train, test)
```

```
## [[1]]
## [1] 0.9948037
##
## [[2]]
## [1] 0.9961538
```

Hence, with the full model we obtain an accuracy of 99% in both train and test dataset. This can be explained by the fact that the `type` of a wine is clearly defined by a combination of its chemical properties, as expected.